Displaying animated GIFs and other images using Qt Quick

This article demonstrates how to use animated GIFs (and other image types containing a series of frames) in Qt Quick.

Introduction

The QML AnimatedImage element makes it very easy to display animations stored as a series of frames in an image (for example an animated GIF file). This article shows how to use this element to animate GIF images, and how to animate the whole image smoothly with respect to its parent.

The example shows a bird flying over a background containing some trees, clouds, sky and sun. The bird flies near the centre of the screen, heading up when the screen is touched and crashing down when the finger/mouse is removed. The implementation is straightforward: the background (everything except the bird) is a single GIF that moves the image across the screen. The bird animated gif is drawn over this. Timers are used to control the rate at which the bird is animated up and down, and these timers are triggered based on a MouseArea that is triggered when the screen is touched. The animation stops when the bird touches the ground.

Flying Bird Animation

Note: The animation is not shown by this image. Try the app to see the animation.

Implementation

Let's create an empty QML project. We display two animated Images, one is the sky and other is the flying bird using AnimatedImage Element.

```
//sky
AnimatedImage
         {
             id: animationCloud; source: "AnimatedClouds.gif"
             height: 555
             paused: false;
         }
//bird
Item {
            id: itemBird
```


(C) Copyright Nokia 2014. All rights reserved.

```
            width: animationBird.width;
            height: animationBird.height + 8
             x: itemBird.width+76;
             y: itemBird.height;
            AnimatedImage {
                id: animationBird; source: "bird-anim.gif"
                paused: false;
                playing: true;
            }
            Behavior on x { SmoothedAnimation { velocity: 200 } }
            Behavior on y { SmoothedAnimation { velocity: 200 } }
    }
```

To make a smooth animation of the flying bird we configure the `velocity` of the SmoothedAnimation. We set two Timers one which increases the *y* position of the flying bird and the other which decreases the *y* position of the flying bird. The timers themselves are triggered based on whether a user is touching the screen MouseArea.

```
Timer {
        id:timerUp
        interval: 100; running: false; repeat: true
        onTriggered:
        {
            if(!crashed)
            itemBird.y = itemBird.y - 10;
        }
    }
```

On release of the MouseEvent we call another Timer which increases the *y* position of the flying bird.

```
  Timer {
        id:timerDown
        interval: 100; running: true; repeat: true
        onTriggered:
        {
            if(itemBird.y >= 470)
            {
                animationBird.paused = true;
                animationCloud.paused = true;
                crashed =  true;
            }
            else if(!crashed)
            {
             itemBird.y = itemBird.y + 20;
            }
        }
    }
```

The result is that the bird flies up when the user touches the screen and down when the touch is removed. The end point of the animation is when the flying bird falls on the ground.

Summary

This article shows how easily we can import Animated Image using QML and change its position smoothly using SmoothedAnimation. It also shows the smooth performance of QML application during multiple active and switching between Timers.

(C) Copyright Nokia 2014. All rights reserved.

Source Code

- The full source code of the example is available here: File:FlyingBirdQML.zip

(C) Copyright Nokia 2014. All rights reserved.